AMENDMENTS TO THE CLAIMS:

This listing of the claims will replace all prior versions, and listings, of the claims in this application.

Listing of Claims:

1. (Currently Amended) A copy engine comprising:

a first register to point to a first address;

a second register to point to a second address, wherein one of the first and second addresses is a source address and one is a destination address for data to be copied;

a control register, wherein the control register comprises,

a count of an amount of memory space required by a copy operation,

an indication of the direction of the copy operation as being one of from the first address to the second address and from the second address to the first address, and

an indication of whether the first address is incremented or decremented,

where the copy engine includes a serialization mechanism in which a write is made to the control register of zero count.

2. (Original) A copy engine as claimed in claim 1, wherein the copy engine includes a locking mechanism for locking the copy engine during a copy operation.

3. (Original) A copy engine as claimed in claim 2, wherein the locking mechanism is locked by a write to the control register and unlocked when the copy operation completes.

4. (Original) A copy engine as claimed in claim 1, wherein a write to the second address triggers the copy operation during which the copy engine is in an active state.

5. (Original) A copy engine as claimed in claim 3, wherein an attempt to write to the control register while the locking mechanism is locked is retried until the current copy operation has

completed.

6. (Original) A copy engine as claimed in claim 4, wherein an attempt to write to any register during the active state is retried until the current copy operation has completed.

7. Cancelled

8. (Previously Presented) A copy engine as claimed in claim 1, comprising a plurality of sets of the first, second and control registers.

9. (Original) A copy engine as claimed in claim 8, wherein each set of registers can carry out copy operations simultaneously and can be locked independently.

10. (Original) A copy engine as claimed in claim 1, wherein an area of unused memory beyond the registers is provided to accommodate a cache line write to the registers.

11. (Currently Amended) A computer system comprising:

a central processing unit in which firmware is stored;

memory in which data is stored; and

a copy engine coupled between the firmware and the memory;

said copy engine comprising,

a first register to point to a first memory address;

a second register to point to a second memory address, wherein one of the first and second memory addresses is a source address and one is a destination address for data to be copied;

a control register, wherein the control register comprises,

a count of an amount of memory space required by a copy operation,

an indication of the direction of the copy operation as being one of from the first address to the second address and from the second address to the first address, and

an indication of whether the first address is incremented or decremented,

where the computer system further comprises a write queue in which waiting copy operations stack up behind a write of zero count and waiting copy operations execute after the write of zero count has completed.

12. Cancelled

13. Cancelled

14. Cancelled

15. Cancelled

16. (Currently Amended) A method of data movement comprising:

maintaining a first register to point to a first address;

maintaining a second register to point to a second address, wherein one of the first and second addresses is a source address and one is a destination address for data to be copied;

operating a control register to:

count an amount of memory space required by a copy operation,

indicate a direction of the copy operation as being one of from the first address to the second address and from the second address to the first address, and

indicate whether the first address is incremented or decremented,

wherein the method includes serialization by making a write to the control register of zero count.

17. (Previously Presented) A method as claimed in claim 16, wherein the method includes locking the copy engine during execution of a current copy operation.

18. (Previously Presented) A method as claimed in claim 17, wherein locking is activated by a

write to the control register and deactivated by completion of the current copy operation.

19. (Previously Presented) A method as claimed in claim 16, wherein a write to the second address triggers the execution of the current copy operation during which the copy engine is in an active state.

20. (Original) A method as claimed in claim 18, wherein an attempt to write to the control register when locking is activated is retried until the current copy operation has completed.

21. (Original) A method as claimed in claim 19, wherein an attempt to write to any register during the active state is retried until the current copy operation has completed.

22. Cancelled

23. (Currently Amended) A method as claimed in claim ~~22~~ 16, wherein at least one waiting copy operation stacks up behind a write of zero count and execute after the write of zero count has completed.

24. (Previously Presented) A method as claimed in claim 16, wherein there are a plurality of sets of first, second and control registers and each set of registers is operable to execute out copy operations simultaneously and is independently lockable.

25. (Previously Presented) A method as claimed in claim 16, wherein the method is carried out by a copy engine that is interposed between the memory and firmware.

26. (Previously Presented) A method as claimed in claim 16, wherein firmware allocates an area of memory as free memory space and initializes the first register to point to the end of the free memory space in memory.

27. Cancelled

28. Cancelled

29. (Currently Amended) A computer program comprising computer executable program instructions stored in a computer readable media, comprising first program instructions to cause a computer agent to issue write operations to store in a copy engine, that is disposed external to the computer agent and coupled to a memory, a count value in a control register for indicating an amount of data to be copied, a first memory address in a first register and a second memory address in a second register, where storing a non-zero count value locks the control register, the first register and the second register from receiving a write operation from another computer agent, and where storing the second memory address initiates execution of a copy operation, further comprising second program instructions to cause the computer agent to issue a write operation to store a count value of zero in the control register to serialize the use of the copy engine with other operations.

30. (Previously Presented) A computer program as in claim 29, where said first program instructions further store in said control register a swap value for enabling a byte swapping operation to be performed on data being copied.

31. (Previously Presented) A computer program as in claim 29, where said first program instructions further store in said control register a copy direction value for specifying a direction that copy data is to flow between the first memory address and the second memory address.

32. (Previously Presented) A computer program as in claim 29, where said first program instructions further store in said control register a value for indicating whether memory addresses are incremented or decremented during the copy operation.

33. (Currently Amended) A computer program as in claim 32, further comprising ~~second~~ other

program instructions that use said value that indicates whether memory addresses are incremented or decremented to maintain at least one stack in the memory.

34. Cancelled

35. (Previously Presented) A computer program as in claim 29, where the source memory address and the destination memory address are 4-byte aligned.

36. (Previously Presented) A computer program as in claim 29, where the write operations are issued in a burst from an internal cache of the computer agent.